

# Cadence Tutorial (Part One)

By Kerwin Johnson

Version: 10/24/05

(based on 6.776 setup by Mike Perrott)

## Table of Contents

Introduction.....	2
Shorthand Conventions.....	4
Getting Help within Cadence.....	4
Troubleshooting.....	4
General Setup.....	5
Setting up Cadence on the MIT Server .....	5
Create a Library.....	6
Schematics and Symbols.....	7
Create the Inverter Schematic.....	7
Create the Inverter Symbol.....	11
Create the Inverter Test Bench.....	13
Create the Device Models.....	15
Simulation.....	19
Configure the Simulator.....	19
Run the Simulation.....	25
Analysis.....	26
Plot Results.....	26
Calculate Values.....	27

## Introduction

This tutorial will introduce the use of Cadence for simulating circuits in 6.012. Cadence is a suite of tools for IC design. It allows for schematic capture, simulation, layout and post-layout verification of analog and digital designs. We will be using a portion of the analog design flow, which can handle up to 200,000 devices.

We will use Composer to enter/capture the schematics of circuits that we will simulate and the test benches with which we will simulate them. We will use Analog Design Environment (ADE) to configure a simulator, in this case Spectre to simulate our circuit with device models that represent the transistors in our circuit.

This tutorial will explain how to set Cadence up on the MIT Server. An overview of the work flow in Cadence is shown in Figure 1. Starting from near the top of the figure we will use the Command Interface Window (CIW) to start the schematic composer. We will capture a schematic of an inverter. We will create a symbol for the inverter. We will create a test bench to test the inverter. We will create device models for our FETs in a text editor. We will open ADE and configure and run our simulation in spectre. Then we will plot our results.

A circuit simulator is a tool. In order to design circuits efficiently you need to use this new tool effectively. The simulator is good at solving thousands of operating points. A good way to use the simulator is to first understand the circuit and then sweep the simulation over the entire area where the solution lies.

We know that the DC transfer function of a well behaved inverter always has a cross over point  $V_M$  where  $v_{in} = v_{out}$ , which we want to find. We will simulate the DC transfer function of the inverter by simulating every point on the curve. Then we can select  $V_M$  by inspection. Because we understand the inverter's behaviour we know that if we sweep the input from 0 to VDD we will find the  $V_M$  if it exists.

It is easy to use the simulator ineffectively. The simulator is not your brain; it can't understand anything about the circuit. If you are just collecting data and not thinking about it, then you are wasting time. The simulator isn't a pen and paper calculation. If you are running single point calculations and aiming for exact agreement with hand calculations you will be frustrated. Instead of running single point simulations and stabbing in the dark, you should think about sweeping some parameter to collect data with trends that you can think about.

The previous three paragraphs were the most important paragraphs of this tutorial. Now we will proceed to become lost in the details of this particular simulator.

## Cadence Workflow

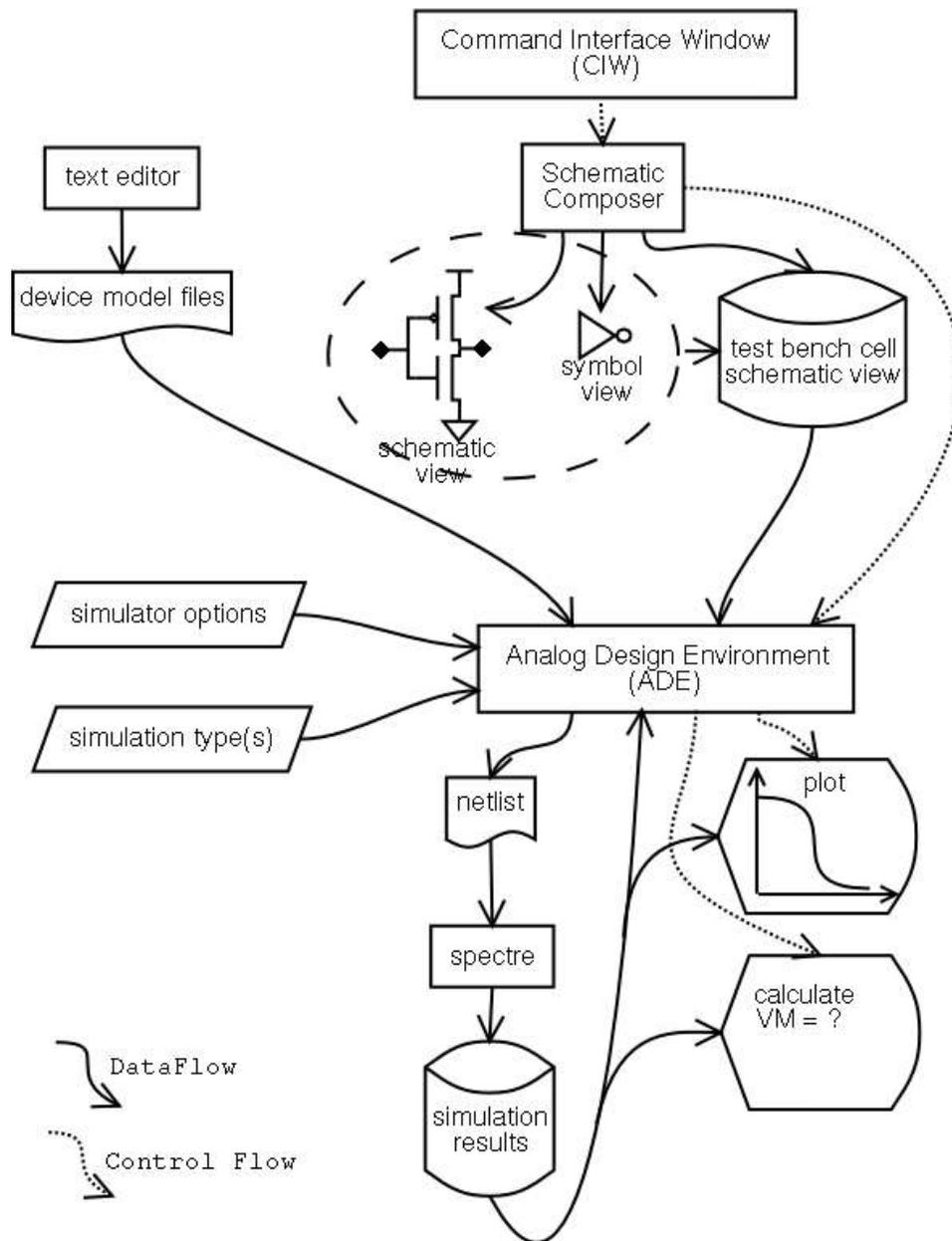


Figure 1 Cadence Design Flow

## Shorthand Conventions

The following are shorthand conventions used in this tutorial

1. When navigating a series of submenus, the tutorial will abbreviate the instruction, "Start in the Command Interface Window (CIW) and click on Tools, then click on Library Manager" to  
Click on CIW->Tools->Library Manager.
2. Cadence has many keyboard shortcuts. When the tutorial writes a letter of a command in parentheses it means that letter is the short cut. Capitalization is significant. For example i is the shortcut for (i)nstantiate.

## Getting Help within Cadence

Here are two ways to get help within the Cadence environment.

1. Click on Help within a Cadence window.  
This will try and start a instance of the Cadence document server cdsdoc.
2. Start the Cadence Document Server from the command line by typing:  
cdsdoc&  
at a unix prompt. The easiest way to use cdsdoc is to click on Search and enter specific terms that you are interested in. Feel free to browse, in order to see the scope of Cadence.

## Troubleshooting

This is a very brief section on trouble shooting. Keep in mind that this is an exercise to configure and run some simple simulations in Cadence. The problems are computer or programming problems. Remember the adage, "Stop the bus. Get out of the bus. Close the doors. Open the doors. Get in the bus. Start the bus." So at any step if you think you have worked Cadence into a weird state. Save your work. Copy it to a new cell, new saved state etc. Exit Cadence if you think it is that hosed. (This is very possible). Go back to the last known good position and start again. The major section heading of this tutorial are good checkpoints.

## General Setup

### Setting up Cadence on the MIT Server

1. Login to the MIT Server on a linux or Solaris machine.
2. Type the following:

```
add 6.012
source /mit/6.012/setup_cadence
```

You can add these lines to your `.cshrc` file in order to have the setup be automatic each time you login.

3. Run Cadence by typing

```
cadence
```

**WARNING:** The first time this is run it will overwrite any `~/cde` directory that you may already have.

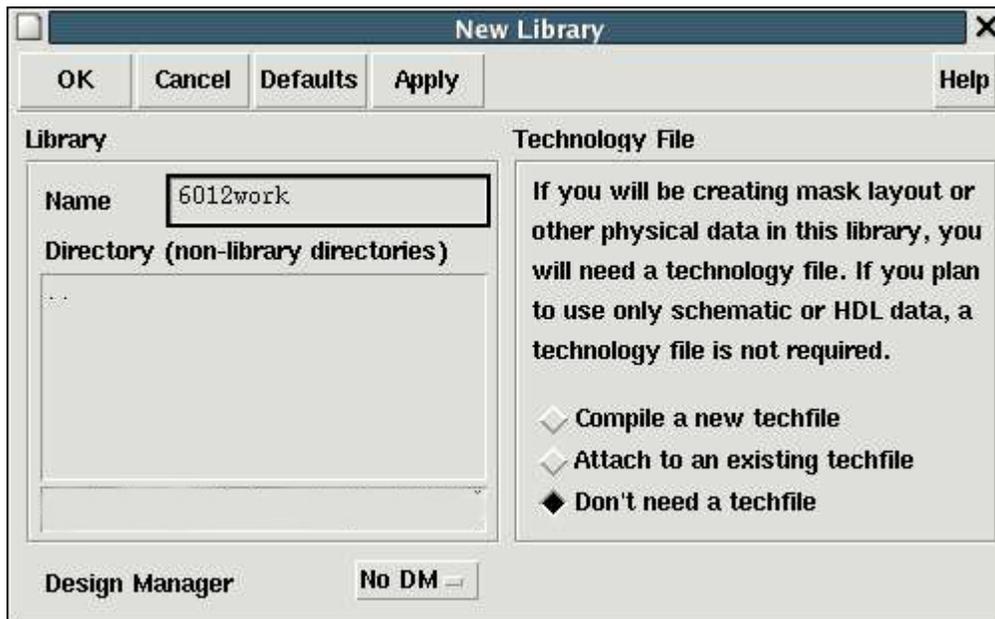
The cadence script starts up icfb (IC Fab) and will open two windows: the CIW (command interface window) and a Library Manager window. If it also opened a What's New in 5.0 window, then close that window.

In the Library Manager you can see that things in Cadence are organised into libraries, cells and cellviews. Libraries are used to categorize cells that belong together. You can look through the analogLib and basic libraries which contain all of the basic analog components that we will use: voltage sources (vdc), current sources (idc), grounds(gnd), resistors (res), capacitors(cap) and other basic things. Cells are names of individual parts/circuits that you can place in a schematic. You can place cells within cells to create a hierarchy. Views/cellviews are ways of looking at a cell. We will use the schematic and symbol views. Close the Library Manager for now.

## Create a Library.

First we need to create a library for our Cadence work.

1. Click on CIW->Files->New->Library. The CIW is the only window left.
2. Change to directory ~/cds/libs. If necessary, create it in unix.
3. Choose a library name.
4. Click on "Don't need a techfile".
5. Click OK.



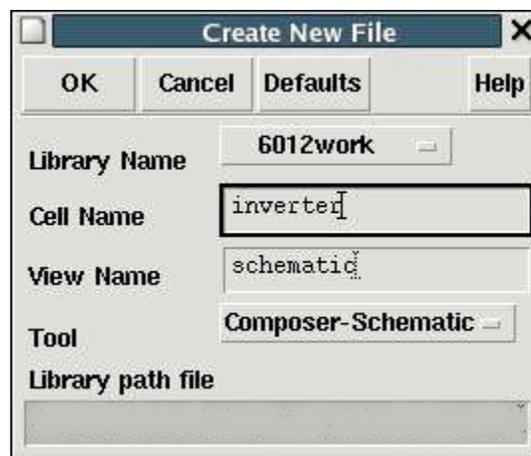
Courtesy of Cadence Design Systems, Inc. Used with permission.

## Schematics and Symbols

### Create the Inverter Schematic.

Now we will open the library manager and create a new schematic.

1. Click on CIW->Tools->Library Manager.  
You should see your new library. If you do not then click on Library Manager(LM)->View->Refresh.
2. Click on your library.
3. Click on LM->File->New->Cellview.
4. Enter the name inverter and the view name schematic and click OK.



Courtesy of Cadence Design Systems, Inc. Used with permission.

This will open a blank Composer schematic capture window. Along the left are buttons that do things. Along the top are pull down menus that do more things. Now is an excellent time to play with all of the buttons and menus, except for the Tools menu. You will notice that many of the command have letters beside them which are their hotkeys, or bindkeys in Cadence parlance.

Before we start here are some useful commands that we will use later. **(i)**nstantiate creates a new object in the schematic. **(p)** creates pins to allow signals to leave the schematic. **(c)**opy allows you to copy objects. **(m)**ove with a lower case m moves objects and drags their wiring with them. **(M)**ove with an uppercase m moves objects after disconnecting their wires. **(l)** allows you to create labels for wires so that they don't get stuck with the default names, like net994875. **(q)**uery opens a dialog box that shows an object properties. **(w)**ire allows you to create wires. After you type w, left click to start a wire, left click to make a corner and double left click to end the wire. Some commands like copy, move and wire have dialog boxes with more options that can be hidden. **(F3)** displays these dialog boxes. Also note that Cadence commands use the bottom line of window as an information bar.

Now we will create the inverter schematic. First we place an NMOS instance from the analogLib library.

1. Press i.

- Click on Browse. Go to the analogLib library and click on the nmos4 cell. Then click on the symbol view. Then click close. nmos4 is a 4 terminal device, which means that the bulk/well terminal is explicitly shown on the symbol.
- We now have a form with all of the options that we can set for an nmos4. Fill it out as shown below.

Courtesy of Cadence Design Systems, Inc. Used with permission.

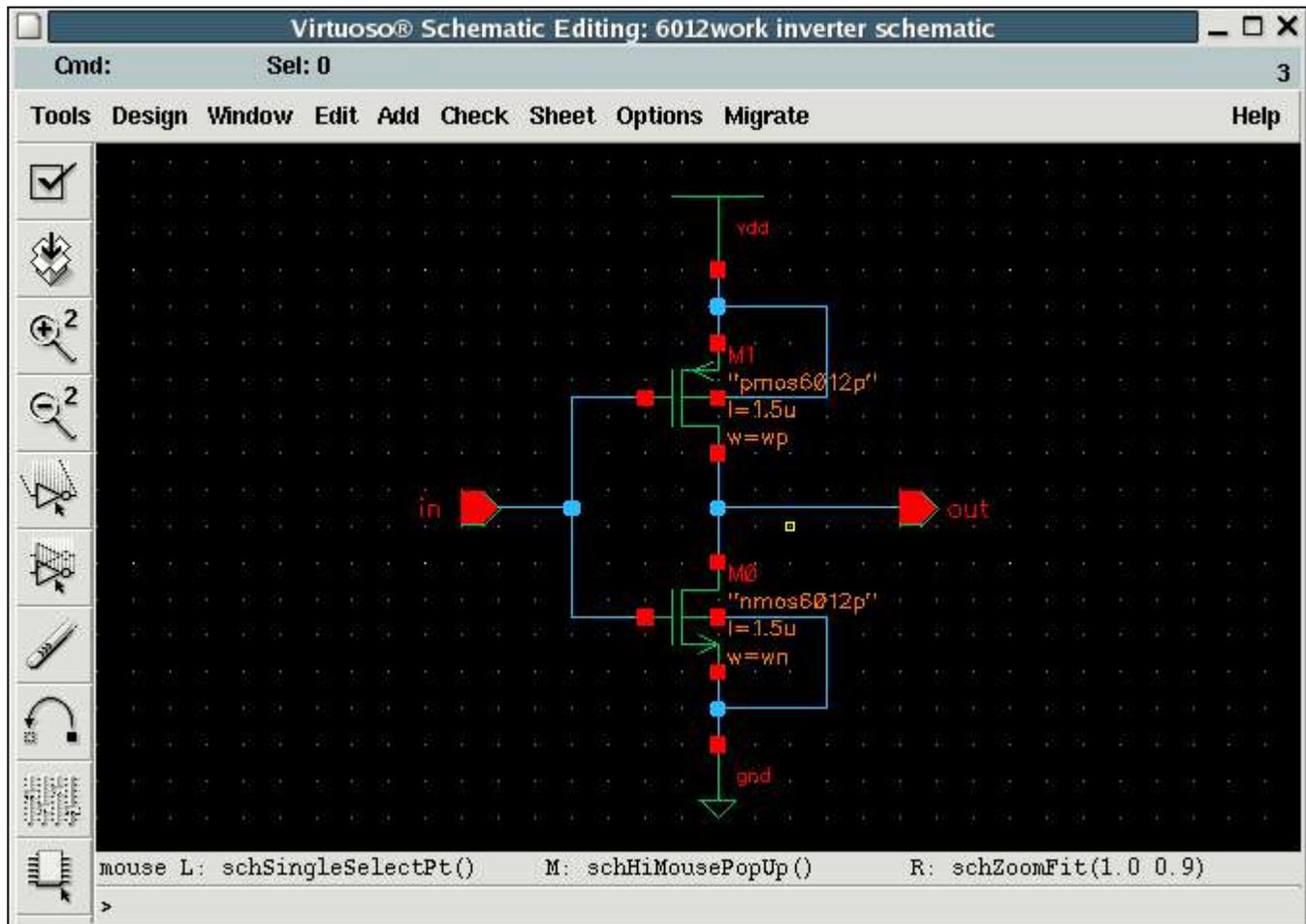
- Now go back to the Composer window and left click somewhere in the black space

to place an instance of the nmos device. Placing an instance uses the value in the form when you left click in the composer window. If you later want to change those values, select the object and press (q)uery to edit the form.

5. Place an **analogLib pmos4 symbol** with parameters model=pmos6012, w=6u, and l=1.5u above the nmos4.
6. Place an analogLib vdd symbol and an analogLib gnd symbol somewhere as well.
7. Type p to create pins. Change the name to in and the type to input. Your form will look like:

Courtesy of Cadence Design Systems, Inc. Used with permission.

8. Create an output pin, called out of type output.
9. Use the (w)ire command by pressing w to create wires to wire everything together. The wiring command works as follows. Left click with the mouse to attach the wire to a pin, a terminal or another wire. Double click to end wire without attaching it to anything. Hit escape to stop wiring. When you finish it should look like:



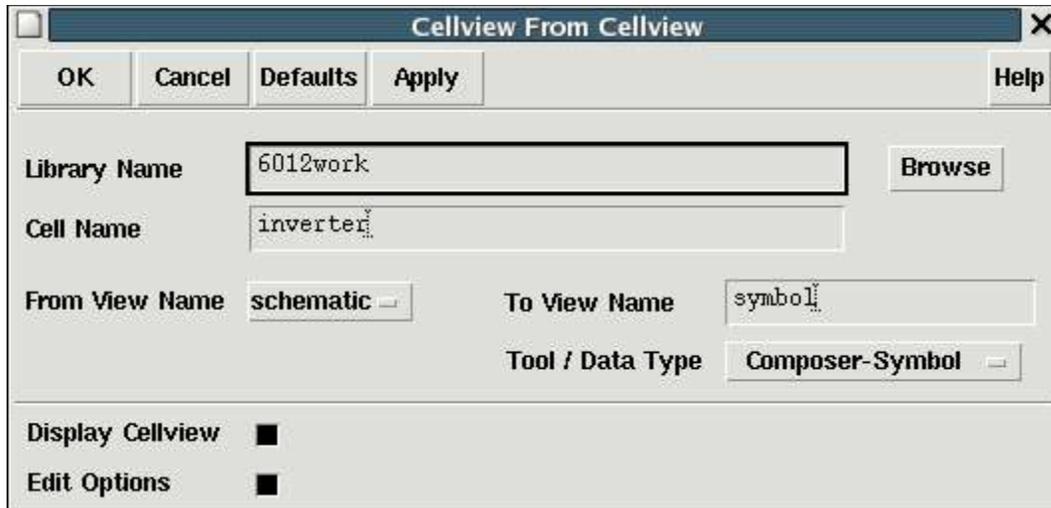
Courtesy of Cadence Design Systems, Inc. Used with permission.

10. To save your schematic press X, or click the check mark right under the Tools menu. This will check and save your schematic. It will report anything it considers an error or a warning. Before you can simulate you need to clear all errors.
11. You are done creating the inverter schematic.

## Create the Inverter Symbol.

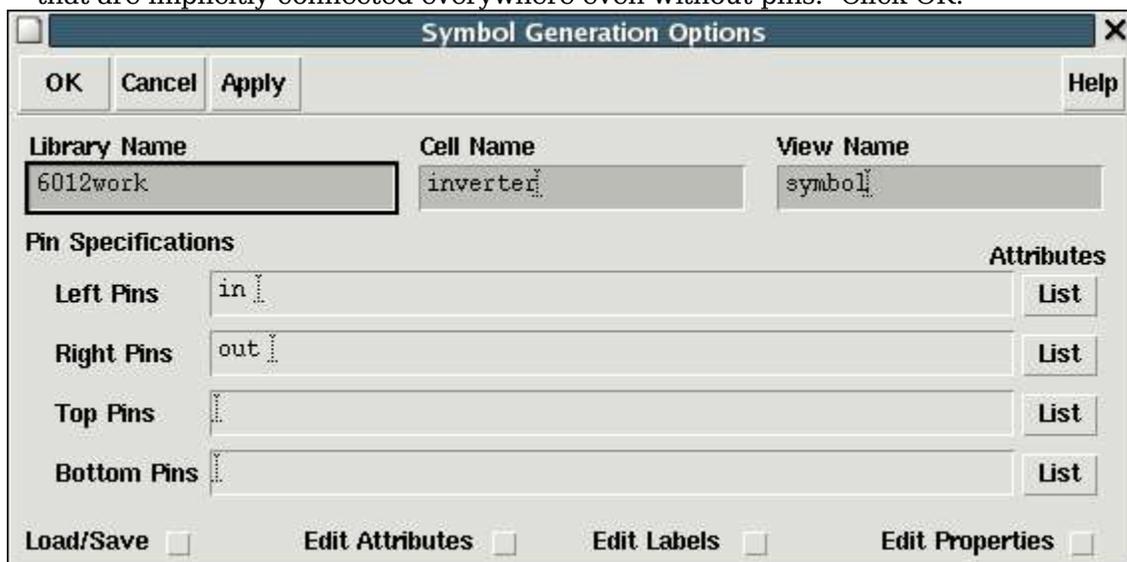
Now that we have a schematic for the inverter we will create a symbol so that we can instantiate our inverter in other schematics without copying the entire inverter schematic each time. We do the following:

1. Click on Schematic Editing->Design->Create Cellview ->From Cellview. Check that the window matches the picture shown below and click OK.



Courtesy of Cadence Design Systems, Inc. Used with permission.

2. Pins can be placed on the left right top or bottom. We want our in pin on the left and our out pin on the right. Shrewd observers will have noticed that there are no pins in the list for vdd and gnd, which will need to be connect to a power supply. The vdd and gnd symbols create nets called vdd! and gnd! which are global nets that are implicitly connected everywhere even without pins. Click OK.

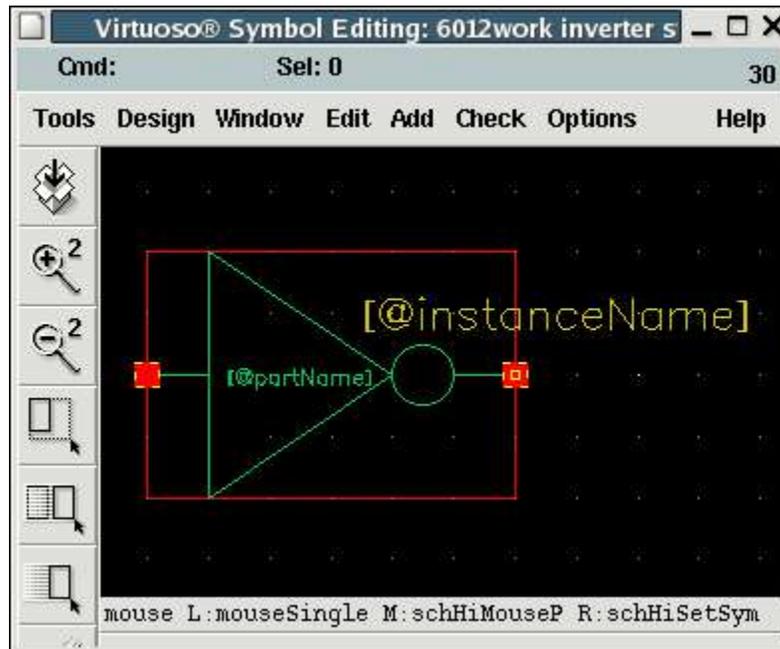


Courtesy of Cadence Design Systems, Inc. Used with permission.

3. Now a symbol editor window will pop open. The default symbol is good enough, but if you want to improve it use the Add->Shape->Polygon, Add->Shape->Line, Add->Shape->Circle, (c)opy, (m)ove/stretch and delete (delete key) commands to make the symbol look as below.

To make the symbol below, you will need to delete the original green box. Grab the edge, stretch it, click in the middle of it and then delete it.

Note the following things. The red bounding box is the area that you will be able to click on in higher level schematics to select your inverter, so don't delete it. If you accidentally delete the pins, it is easier to go back to the inverter schematic and start over.



Courtesy of Cadence Design Systems, Inc. Used with permission.

4. Remember to (S)ave your symbol before you exit

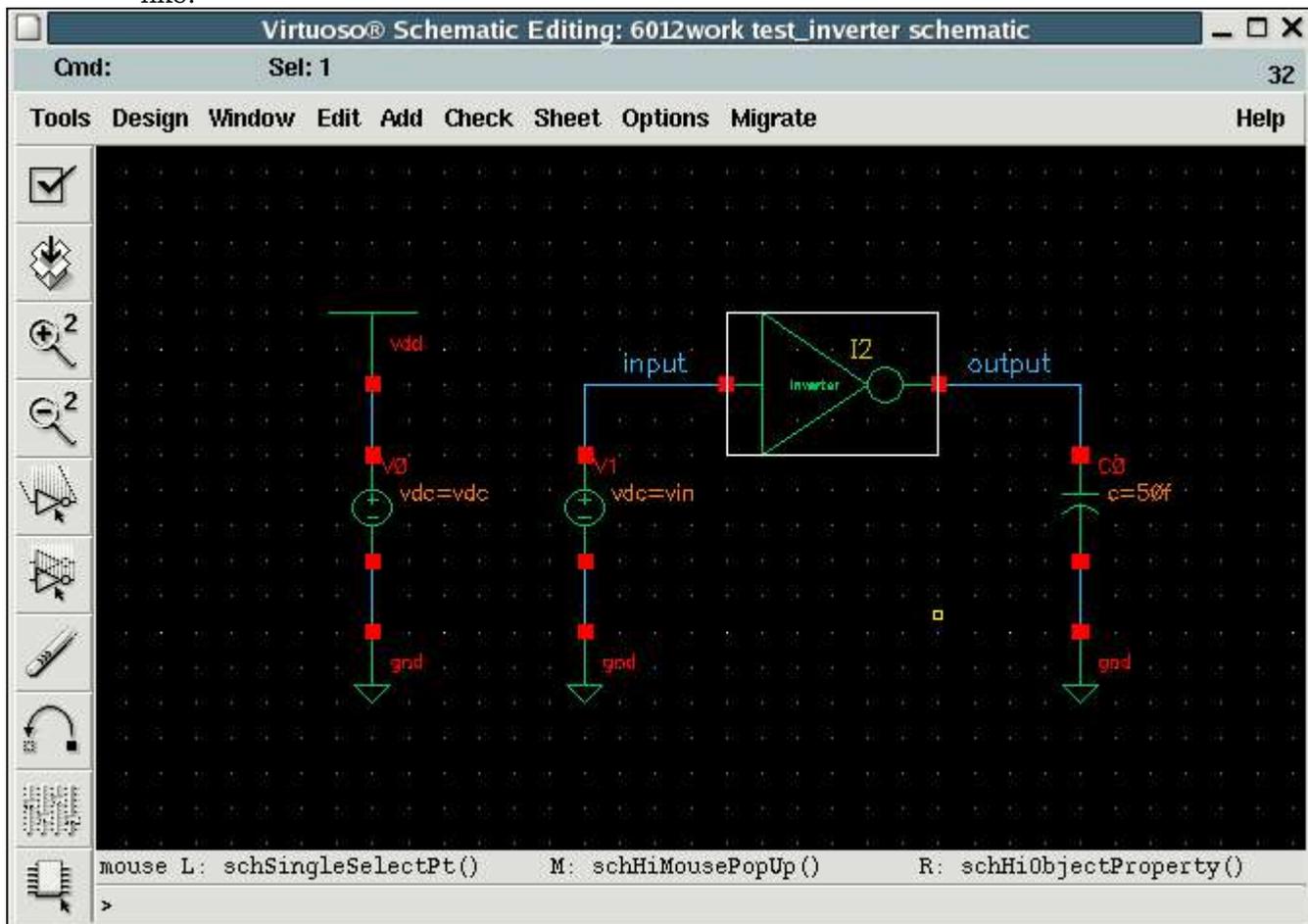
## Create the Inverter Test Bench.

Now that we have an inverter we will want to test it. We will create an entirely new schematic to test the inverter. You can think of this schematic as a testbench in the lab. We want to create a test bench to measure the DC transfer curve of the inverter. To do that we will need a VDD supply for the inverter, a input signal for the inverter and an appropriate output load for the inverter.

1. Create a schematic called test\_inverter in your 6012work library.
2. Instantiate two of analogLib vdc symbol, three analogLib gnd symbol, an analogLib vdd symbol, one analogLib cap symbol and one of your work6012 inverter symbol. Use (q) to set the voltage of the power supply DC voltage (3<sup>rd</sup> box) to vdc, the input supply to vin and the load capacitor to 50f(1<sup>st</sup> box).

Setting the power supply and input supply to the words vdc and vin means that they are variables that will be set in the simulator. This is a powerful technique to simulate many states without changing your schematic every time. You can use it with most fields of most symbols. You can also make arbitrary equations using Cadences built in scripting language Skill (which is very similar to Scheme).

Use (l) to label the wires. If you want to descend into the inverter use (E)nter to descend and Ctrl-e to ascend or exit the inverter. After you are done it should look like:



3. Check and save (X).

## Create the Device Models.

In order to simulate our inverter we will need models for the NMOS and PMOS devices that we are using. In the real world if you are a circuit designer you are probably provided a set of models from the fabrication plant. If you are a device engineer you make these models.

We will be using the spectre mos level 1 models, called mos1. There are about 40 standard MOS models in Cadence and hundreds of non-standard models in industry. All models have parameters which can be set within the model and will be the same for every device of that type. They also have instance parameters which are set individually for every device.

Below are tables of these parameters for the MOS1 model in spectre, their defaults and the values that we want to use in 6.012. Every MOS will have an individual length (l) and width (w).

Parameter Name	Spectre Name	Default Value	NMOS	PMOS	Units
Model Name		mos1	mos1	mos1	none
Device Type	type	n	n	p	none
Zero bias threshold voltage	vto	0	0.75	-0.75	V
Transconductance (mu Cox)	kp	2.07E-005	1.00E-004	5.00E-005	A/(V <sup>2</sup> )
Channel length modulation parameter	lambda	0	7e-2*1.5u/length	7e-2*1.5u/length	V <sup>-1</sup>
Surface potential	phi	0.7	0.7	0.7	V
Body effect parameter	gamma	0	0.6	0.6	sqrt(V)
Carrier Surface Mobility	uo	600	600	600	cm <sup>2</sup> /(V s)
Carrier Saturation Velocity	vmax	infinity	infinity	infinity	m/s
Mobility Modulation Coefficient	theta	0	0	0	1/V
Oxide Thickness	tox	1.00E-007	1.50E-008	1.50E-008	m
Zero-bias planar bulk depletion capacitance	cj	0	1.00E-004	3.00E-004	F/(m <sup>2</sup> )
Zero bias sidewall bulk depletion capacitance	cjsw	0	5.00E-010	3.50E-010	F/m
Bulk Junction built in potential	pb	0.8	0.9	0.9	V

Parameter Name	Spectre Name	Default Value	NMOS	PMOS	Units
Planar bulk junction grading coefficient	mj	0.5	0.5	0.5	none
Sidewall bulk junction grading coefficient	mjsw	0.33	0.33	0.33	none

Table 1MOS1 Model Parameters

Each time that you create a MOS device in a schematic it is called an instance. The following table lists all of the instantiated parameters that can be changed for each instance or device that you create.

W and L are in this table and for that we happy. Think about and write formulae in terms of  $l$ ,  $w$  and  $ldiff$  (the length of the source/drain diffusion) for the diffusion area ( $lds$ ,  $ldd$ ), the diffusion perimeter and number of squares of the source and drain to fill in the empty values on the table. You will need to change these every time that you change the  $w$  and  $l$  device sizes. We will look at a more sophisticated model in the second tutorial that automatically calculates expected values for these.

You may have noticed that  $\lambda$  is in the non-instantiated parameter table and we want it to depend on length. Our more unsophisticated model will also overcome this problem.

Instance Parameter Name for mos1	Spectre Name	Default Value	NMOS	PMOS	Units
Channel width	w	3.00E-006	3.00E-006	3.00E-006	m
Channel Length	l	3.00E-006	1.50E-006	1.50E-006	m
Area of Source Diffusion	as	0	?	?	m <sup>2</sup>
Area of Drain Diffusion	ad	0	?	?	m <sup>2</sup>
Perimeter of Source Diffusion	ps	0	?	?	m
Perimeter of Drain Diffusion	pd	0	?	?	m
Number of Square of Source diffusion	nrs	0	?	?	none
Number of square of drain diffusion	nrd	0	?	?	none

<b>Instance Parameter Name for mos1</b>	<b>Spectre Name</b>	<b>Default Value</b>	<b>NMOS</b>	<b>PMOS</b>	<b>Units</b>
Length of Source Diffusion	lds	0	4.50E-006	4.50E-006	m
Length of Drain diffusion	ldd	0	4.50E-006	4.50E-006	m

Table 2MOS1 Model Instance Parameters

To create the model file do the following:

1. If necessary create a directory `~/cds/models/`
2. Create a file called `mos6012.scs` that contains the following:

```
// Created by: Kerwin Johnson Sept 2005.
```

```
simulator lang=spectre insensitive=yes
```

```
model nmos6012 mos1
```

```
+ type = n
```

```
+ l = 1.5e-6
```

```
+ w = 3.0e-6
```

```
+ vto = 0.75
```

```
+ kp = 100e-6
```

```
+ lambda = 7e-2
```

```
+ phi = 0.7
```

```
+ gamma=0.6
```

```
+ tox = 1.5e-8
```

```
+ cj = 1e-4
```

```
+ cjsw = 5e-10
```

```
+ pb = 0.9
```

```
+ lds = 4.5e-6
```

```
+ ldd = 4.5e-6
```

```
+ as = 1.35e-11
```

```
+ ad = 1.35e-11
```

```
+ ps = 1.2e-5
```

```
+ pd = 1.2e-5
```

```
+ nrs = 1.5
```

```
+ nrd = 1.5
```

```
model pmos6012 mos1
+ type = p
+ l = 1.5e-6
+ w = 6.0e-6
+ vto = -0.75
+ kp = 50e-6
+ lambda = 7e-2
+ phi = 0.7
+ gamma=0.6
+ tox = 1.5e-8
+ cj = 3e-4
+ cjsw = 3.5e-10
+ pb = 0.9
+ lds = 4.5e-6
+ ldd = 4.5e-6
+ as = 2.7e-11
+ ad = 2.7e-11
+ ps = 1.5e-5
+ pd = 1.5e-5
+ nrs = 1.5
+ nrd = 1.5
```

### 3. Save the file.

The model file that you have created, tells Spectre the following: Create a model called nmos6012 of type mos1. mos1 is the second simplest of ~40 standard CMOS differential equation models. Set the default values (l, w, vto etc.) as indicated. Whenever an nmos6012 is instantiated it may have any of the instance parameters in Table 2 override the default values.

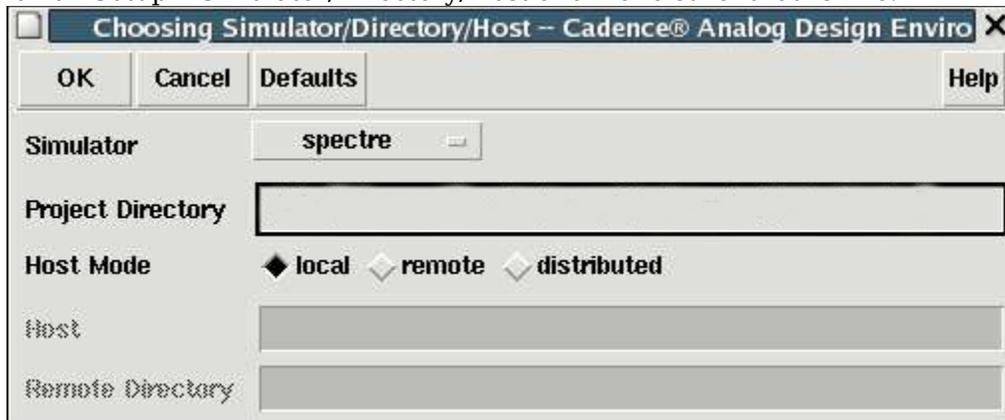
## Simulation

### Configure the Simulator.

In this section we are going to start ADE and use it to configure the simulator, Spectre to run a DC simulation and trace the transfer function of the inverter. A DC simulation sets the DC/constant/large signal values of some nodes, like the supply and the input. It then calculates all of the other values. To see a transfer function we will sweep the input of the inverter from 0 to VDD volts.

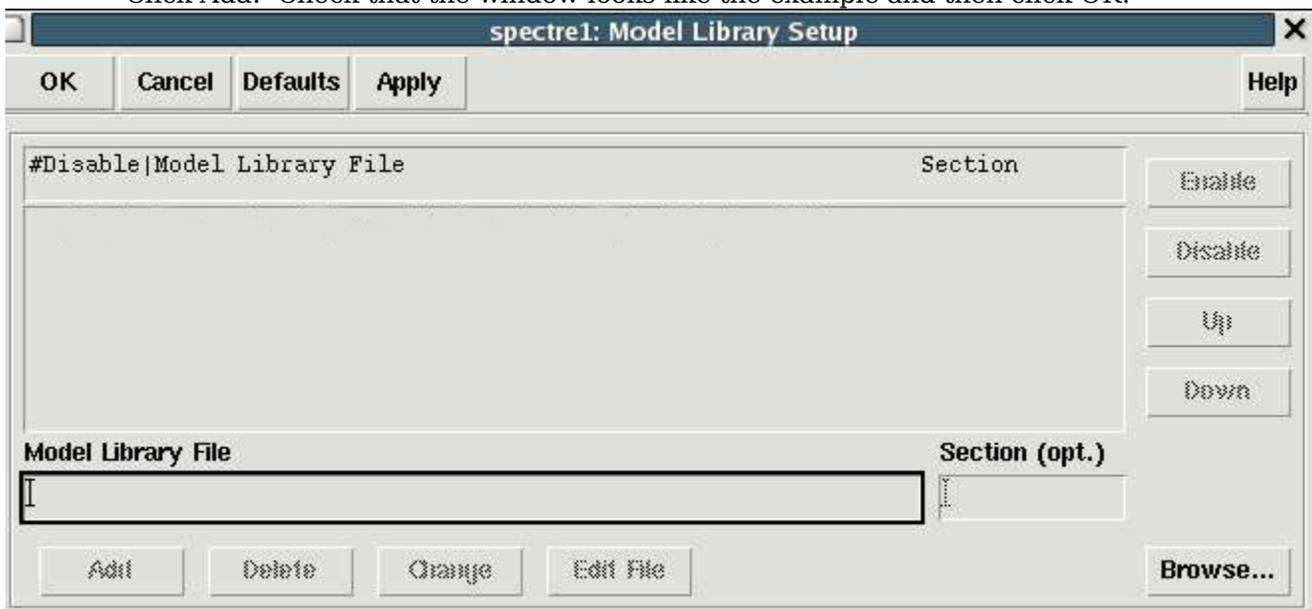
Lets get going:

1. Start ADE by opening the schematic test\_inverter and clicking on Tools->Analog Environment. Now we will set up ADE which is common to all types of simulations.
2. Click on Setup->Simulator/Directory/Host and make sure it looks like.



Courtesy of Cadence Design Systems, Inc. Used with permission.

3. Add the models that we just created. Click on Setup->Model Libraries. Click Browse. Go to the ~/cds/models/ directory and select mos6012.scs. Click OK. Click Add. Check that the window looks like the example and then click OK.



Courtesy of Cadence Design Systems, Inc. Used with permission.

4. Click Setup->Temperature and change the temperature to 25 degrees and click OK.
5. Every time that you setup a simulation, where you do not load a previous state you will have to repeat these steps, unless you find out how to save these as your Cadence default. If your TA discovers that you have changed your Cadence defaults and that is the source of your woes, it will go badly for you.

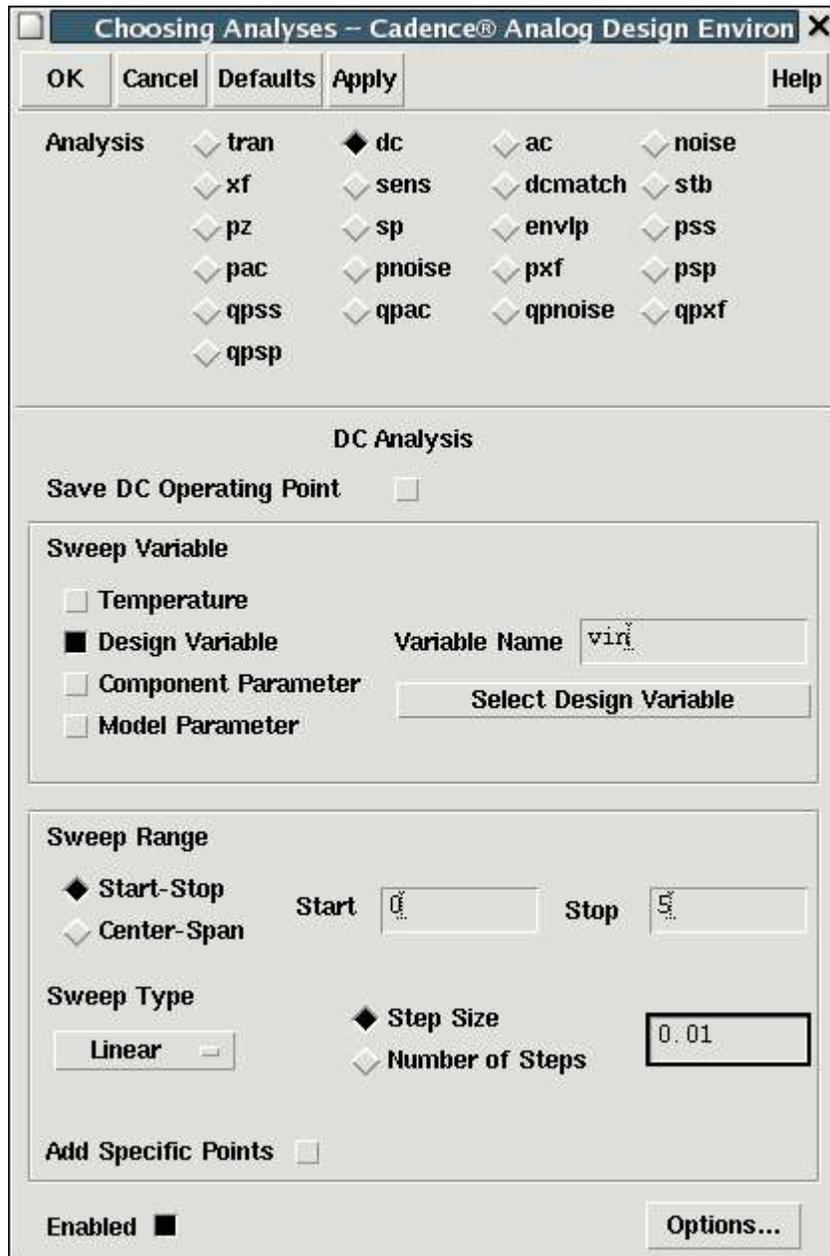
Now we will setup a DC simulation where we sweep in input from 0 to 5 volts.

1. Set the design variables vdc and vin to default values. Click Variables->Edit. In the form type that name of the variable vdc, the value 5 and click add. Then set vin to 2.5. Click OK. The form should look like:

Selected Variable		Table of Design Variables		
Name	Value (Expr)	#	Name	Value
		1	vdc	5
		2	vin	2.5

Courtesy of Cadence Design Systems, Inc. Used with permission.

2. Create a DC simulation where you sweep the input voltage. Click Analyses->Choose. Select dc. Select from within Sweep Variable the Design Variable button. Set the variable name to vin. Set Start to 0 and Stop to 5. Set Sweep Type to linear. When stepsize appears set it to 0.01. Check the example and then click OK.



Courtesy of Cadence Design Systems, Inc. Used with permission.

3. Set the simulator to save everything. Click on Outputs->Save All. Select save = allpub, current = all, then click OK.

In general this is a bad idea, because the simulator can generate gigabytes of data, which may cause the simulating machine to crash. Furthermore, writing all of that data slows down the simulation and the network. So, if you know what data you want to look at, then only save that data. We have a very small circuit so we will save everything.



Courtesy of Cadence Design Systems, Inc. Used with permission.

4. Check the simulator options. Click on Simulation->Options->Analog.

These are all of the options that you can pass to any analog simulator. You may have noticed an options button on the DC simulation setup which leads to a similar form with options specific to the DC simulation. The `reltol`, `vabstol` and `ibastol` and the most important analog options. `vabstol` and `iabstol` are the absolute maximum allowable difference in voltage or current at given node or terminal between two time steps. The simulator re-computes all of the difference equations every timestep until every point falls below this threshold. Except for one major exception. Spectre will accept a solution if the relative error between solutions in two different time steps is less than `reltol`. This can be a problem if your simulation has large values such as 1kV with small signals less the `reltol*1kV` superimposed.

You should not need to change anything in this window for 6.012. Click Cancel.

5. Manually create the Netlist. You usually only need to do this step if something odd is happening or you need to hand tweak the netlist. Click Simulation->Netlist->Create. Look at the netlist.

This is the computer generated file that is passed to the spectre simulator when we run our simulation. Reading through it we see the following. The simulator is told to expect instructions in spectre syntax. The global nodes for ground and vdd are created. The parameters that we will sweep, `vin` and `vdc` are created. The model file that we made is included. A sub-circuit of our inverter is listed. Since it is a subcircuit we can include the exact same circuit many times in the overall file. Notice that the syntax for a mos device is:

Mxxxx (Drain Gate Source Bulk) modelname parameters

Continuing through the file we see the testbench schematic. Then there are the simulator options that we glanced at. The line that start with dc creates the dc simulation. The remainder of the file indicates what data to save.

Check that your netlist matches the example. Close this window.



```
// Generated for: spectre
// Generated on: Sep 24 22:18:07 2005
// Design library name: 6012work
// Design cell name: test_inverter
// Design view name: schematic
simulator lang=spectre
global 0 vdd!
parameters vdc=5 vin=2.5
include

// Library name: 6012work
// Cell name: inverter
// View name: schematic
subckt inverter in out
    M1 (out in vdd! vdd!) pmos6012 w=6u l=1.5u
    M0 (out in 0 0) nmos6012 w=3.0u l=1.5u
ends inverter
// End of subcircuit definition.

// Library name: 6012work
// Cell name: test_inverter
// View name: schematic
C0 (_net1 0) capacitor c=50f
I2 (_net0 _net1) inverter
V0 (vdd! 0) vsource dc=vdc type=dc
V1 (_net0 0) vsource dc=vin type=dc
simulatorOptions options reltol=1e-3 vabstol=1e-6 iabstol=1e-12 temp=25.0 \
    tnom=27 scalem=1.0 scale=1.0 gmin=1e-12 rforce=1 maxnotes=5 maxwarns=5 \
    digits=5 cols=80 pivrel=1e-3 ckptclock=1800 \
    sensfile="..psf/sens.output"
dc dc param=vin start=0 stop=5 step=0.01 write="spectre.dc" \
    oppoint=rawfile maxiters=150 maxsteps=10000 annotate=status
modelParameter info what=models where=rawfile
element info what=inst where=rawfile
outputParameter info what=output where=rawfile
designParamVals info what=parameters where=rawfile
saveOptions options save=allpub currents=all
```

6. If you are happy with your simulation setup then save it. Click Session->Save State. Give it name and decide what you want to save (everything) and click OK. You can use this after you have plotted data, or created equations to save some or all of the setup that created those results. This doesn't save your data, just the simulation setup.

## Run the Simulation.

In this section we complete the long and arduous procedure for running a simulation. Start at step 1.

1. Click Simulation->Netlist and Run. Or Click the green stoplight.

This creates a new netlist from the schematic and runs the simulation. If you do Simulation->run or click the yellow stoplight ADE will re-run a netlist that already exists. This is how you hand tweak and run simulations from with ADE. Note that the schematics need to be Checked and Saved, not just saved in order to simulate. If you are playing with the model files, then you will need to Check and Save to schematics in order to convince ADE to re-netlist.

If you have problems, look at your netlist and compare it with the one shown above.

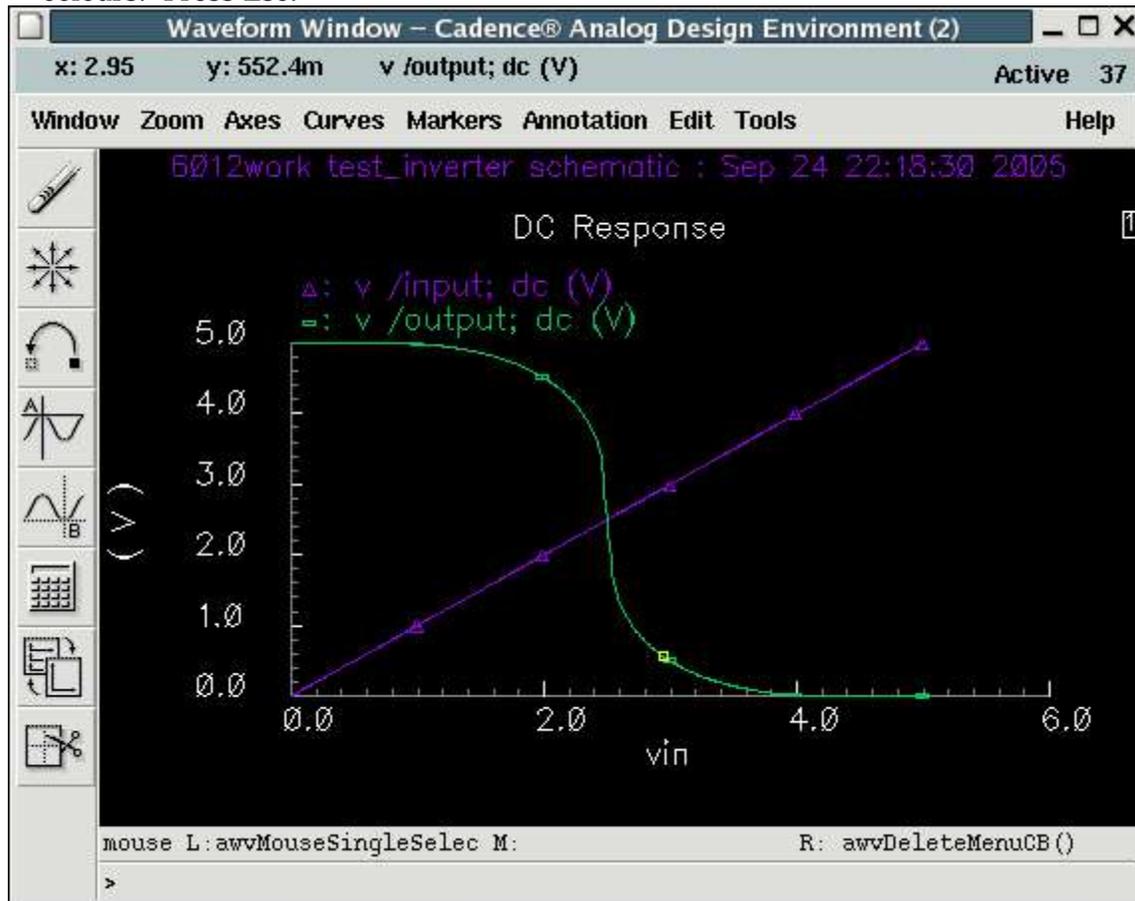
## Analysis

### Plot Results.

Here we will plot the DC transfer function of the inverter and calculate the crossover voltage where  $v_{in} = v_{out}$  and calculate the gain at the crossover voltage.

First lets plot the DC transfer function

1. Click Results ->Direct Plot->Main Form.
2. Select the output and the input wires on the schematic. They should turn different colours. Press Esc.



Courtesy of Cadence Design Systems, Inc. Used with permission.

3. Titles and labels can be changed under Annotation. You can print this to an eps file under Window->Hardcopy.

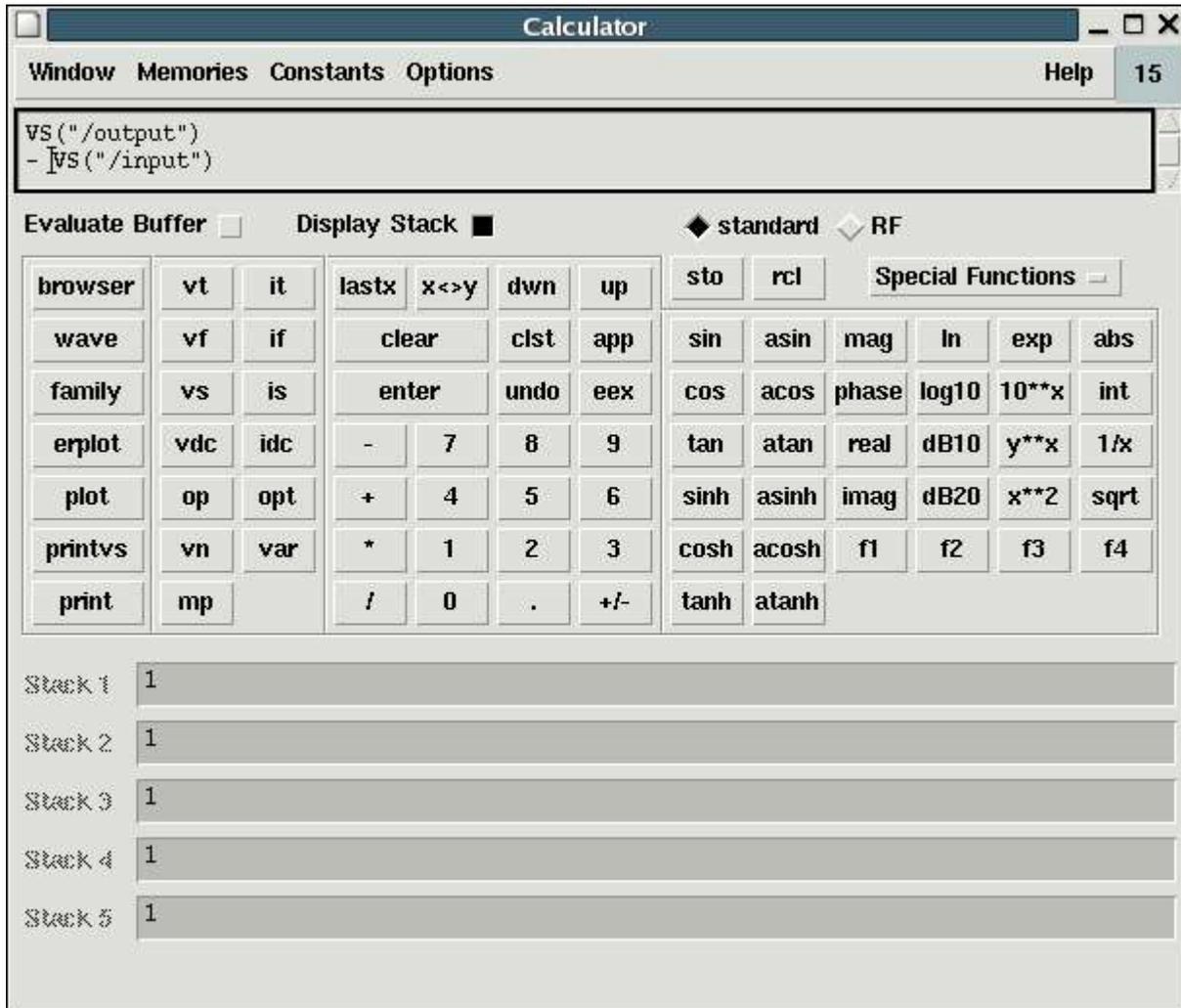
## Calculate Values.

Now we will calculate the crossover voltage. We can see it clearly on the plot. We will open the calculator and find where  $v_{in} = v_{out}$ . One way to do this is to find where  $v_{out} - v_{in}$  crosses zero.

The calculator allows you to make expressions of the data from the simulation. We can do arithmetic on and calculate common parameters of interest to designers.

1. Open the calculator. Click Tools->Calculator. Turn on Display Stack (middle of the window). The calculator is an RPN calculator and viewing the stack makes it easier to understand what is happening.
2. Form the expression  $V_{out} - V_{in}$ . The buttons in the second column from the left refer to types of data: vt,it (transient data voltage and current), vdc,idc (single point dc operating point) and vs,is (dc sweep data).
  - a) Click vs.
  - b) Click on the output wire in the schematic window.
  - c) Click on the input wire on the schematic window. Notice that the VS ("/output") in the calculator was pushed onto the stack.
  - d) Click - (minus) on the calculator. Notice the expression in the calculator window is  
$$VS("/output") - VS("/input")$$

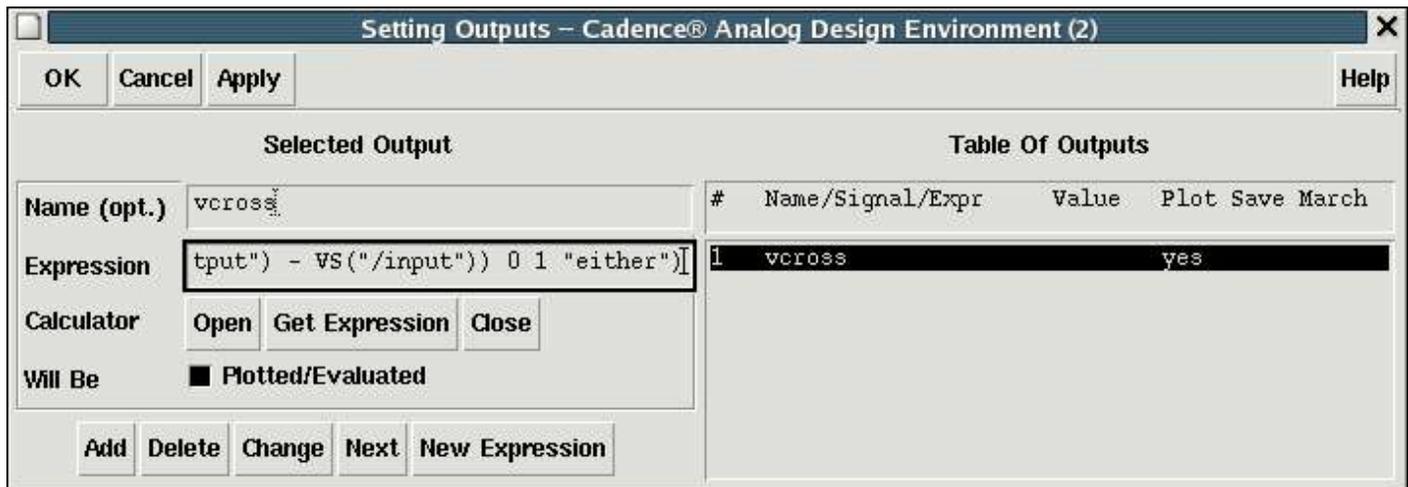
You could plot this if you want, by clicking plot in the far left column.
3. Calculate the crossover voltage. Click Special Functions(near right side) ->cross. Set the threshold value to 0 and click OK.



Courtesy of Cadence Design Systems, Inc. Used with permission.

- You could plot this if you want, but as it is a single data point it makes more sense to display it in the output section of the ADE window. Plot it and then delete it to make sure that your expression is correct. There needs to be a space after the minus sign.

Click ADE->Outputs->Setup. Call it vcross. Click Get Expression. Leave plotted/evaluated selected so that it recomputes every time you run the simulation. Click Add. Click on the word vcross in the Table of Outputs frame. It should look like the picture. Click OK.

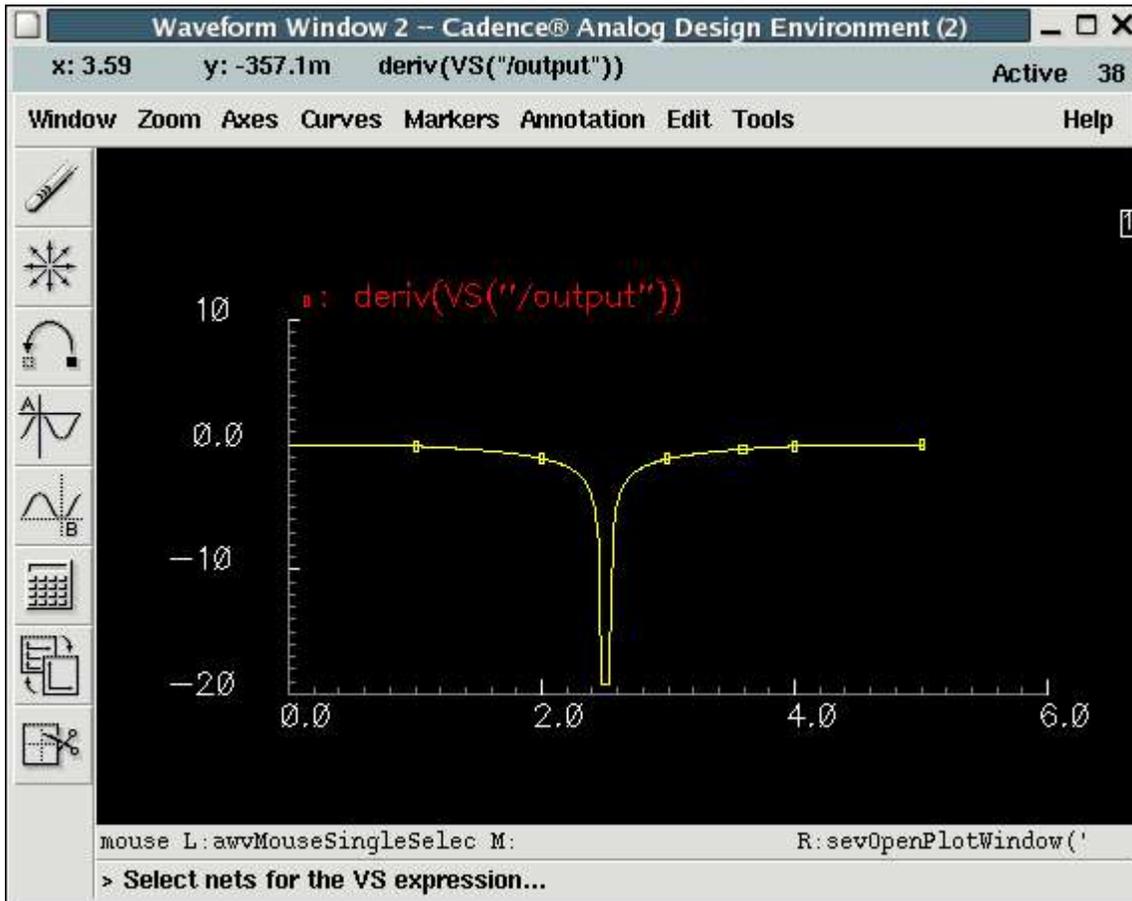


Courtesy of Cadence Design Systems, Inc. Used with permission.

5. Now every time you run the simulation it will calculate vcross. Force it to calculate it now by clicking ADE->Results->Plot Outputs->Expressions.

Now that we have calculated the crossover voltage, we will calculate the gain and the gain at the crossover voltage. Gain in this case is the slope of the sweep output curve VS("/output").

1. With VS("/output") in the Calculator window select Special Functions->deriv.
2. Plot this by opening a new waveform window (ADE->Tools->Waveform) and then clicking Calculator->plot. You can notice from the plot that the gain of the inverter peaks very sharply at one voltage. We hope that the voltage is the crossover voltage.



Courtesy of Cadence Design Systems, Inc. Used with permission.

3. Now calculate the gain at the crossover voltage. Click Calculator->Special Functions->value. Put 2.5 in the box. Click OK. The expression in the calculator should be  $\text{value}(\text{deriv}(\text{VS}("/\text{output}")), 2.5)$ .
4. Add this to your ADE->Outputs window (ADE->Setup->Outputs).
5. For bonus and to save yourself time in the near future figure out how to automatically calculate
  - a) the gain at  $v_{\text{cross}}$  automatically
  - b) the peak gain.
6. Now is good time to save your ADE state. ADE->Session->SaveState. This will mean that you can reuse your expressions and setup on another circuit.

Play time.